REMARKS

Claim Status

Claims 25-39 are pending in the application. This paper does not amend the claims. Claims 25, 32, and 35-38 are the independent claims of the application.

Art Rejections

The Final Office Action rejected claims 25-35 and 37 under 35 U.S.C. § 102(e) as being anticipated by Sekido, U.S. Patent Number 6,311,193 ("Sekido" hereinafter). Claims 36, 38, and 39 were rejected as being unpatentable over Sekido in view of Rungta, U.S. Patent Number 6,484,186 ("Rungta" hereinafter). Applicants respectfully request reconsideration of these rejections based on the following arguments.

Applicants have previously argued that Sekido does not disclose a summary map that refers to a file with an inclusive OR'ed bitmap of the vacancy bitmaps of the snapshots maintained by the file system. In response to this argument, the Final Office Action asserts that the rejected claims do not recite the inclusive OR'ed bitmap limitation, and that "[a]lthough the claims are interpreted in light of the specification, limitations from the specification are not read into the claims." In the present case, however, the inclusive OR'ed bitmap limitation is included in the claims through the recitation of *summary map*, as this term is defined in the specification.

It is well established that an applicant can be his or her own lexicographer. MPEP § 2173.05(a). "When the specification states the meaning that a term in the claim is intended to have, the claim is examined using that meaning, in order to achieve a complete exploration of the applicant's invention and its relation to the prior art." *Id.* (citing *In re Zletz*, 893 F.2d 319 (Fed. Cir. 1989)). Here, the Applicants have exercised the right to define the meaning of the expression *summary map*. This expression is defined in the specification: "Summary map — In general, the term 'summary map' refers to a file including an IOR (inclusive OR) bitmap of all the snapmaps." Specification, at page 11, lines 4-5 (emphasis added). The term *snapmap*, which is used in the definition of *summary map*, is also defined in the specification: "Snapmap — In general, the term 'snapmap' refers to a file including a bitmap associated with the vacancy of blocks of a snapshot. The active map diverges from a snapmap over time as the blocks used by the active file system change during consistency points." Specification, at page 10, line 21 through page 11, line 2. Thus, a *summary map* is an inclusive OR bitmap of the vacancy bitmaps of the snapshots maintained by the file system.

Indeed, the definitions of *summary map* and *snapmap* are given under the *Lexicigraphy* heading in the Detailed Description of the present application, hailing to the multitude of decisional and other authority that allows the Applicants to be their own lexicographers. The definitions may not be ignored in determining the scope of the claims. The claims in issue should be interpreted using the meaning of the term *summary map* as this term was expressly defined in the present application. Because Sekido fails to disclose the inclusive OR'ed bitmap limitation, Sekido does not anticipate independent claims 25, 32, 35, 36, and 37.

In responding to the Applicants' arguments regarding the *summary map* limitation, the Final Office Action also asserted that

> a summary map is shown in Fig. 31. This summary map comprises two copies of earlier active map ST1 and ST3 included in at least two of said snapshots (SS1 and SS3). It should be noted that the snapmaps ST1 and ST3 represent the snapshots SS1 and SS3. As seen in Fig. 31, the summary map and snapmaps ST1, ST3 indicating the valid blocks (i.e. free blocks) B2, B7, B8, B12, and B13 and the invalid blocks (i.e. in-used blocks) Bl8, Bl9, B20, B21, B22, B24, B25, B27, B29.

Final Office Action, at 10. This statement apparently confuses stripes ST1 and ST3 with *snapmaps*.

The full definition of a *snapmap* in the present application has already been quoted above. It is "a file including a bitmap associated with the vacancy of blocks of a snapshot." Specification, at 10, lines 21-22. A *snapshot* is also defined in the present application. It is "a copy of the file system." Specification, at 10, line 16. Thus, a snapmap is a file including a bitmap associated with the vacancy of blocks of a copy of the file system.

In contrast, Sekido's ST1 and ST3 are stripes of file data. According to Sekido, "[d]ata is written into the disk device 4 in predetermined units called stripes, each stripe being an integral multiple (K) of the block size." Sekido, col. 6, lines 18-20. A stripe is created as follows:

> Then, when the data items received from the file system 2 have amounted to a stripe of data items less one block (or a K-1 number of data items) in the writing buffer 8, the disk snapshot control section 5 writes them into the disk device 4. At this time, the disk snapshot control section 5 creates a logical address tag block from the logical address for each block stored in the buffer management table 9 and the time stamp on the volatile memory 6. (The address data and data block in the logical address tag block are forced to have a one-to-one correspondence beforehand, which helps identify the logical address for each block.) Thereafter, the disk snapshot control section 5 writes one stripe of data items added with the logical address tag block into the empty area of the disk device 4 (see FIG. 7).

Sekido, col. 6, line 66 through col. 7, line 12. Thus, each stripe contains no more data than would fit in the writing buffer. Sekido's disclosure leaves little doubt that the size of the stripe (writing buffer) is generally smaller than a snapshot. Consider, for example, Sekido's Figure 33, which shows that each snapshot may contain data stored in multiple stripes.

Moreover, Sekido appears to acknowledge that ST1 and ST3 need not be snapshots or snapmaps: "When there is a snapshot, simple repacking cannot empty the area to be repacked." Sekido, col. 17, lines 63-64. Repacking here apparently refers to the repacking or integrating of ST1 and ST3 into ST10, which procedure Sekido describes immediately above the quoted statement. Sekido then describes various ways to "overcome this problem" of not being able to empty the area by repacking. See Sekido, at column 17, line 66 through column 19, line 27. By pointing out that the repacking procedure illustrated in Figure 31 may not be carried out when there is a snapshot, Sekido apparently acknowledges that the procedure may be carried out when the is no snapshot. Without snapshots, there are no snapmaps. Thus, ST1 and ST3 need not be snapmaps.

Stripes ST1 and ST3 are not bitmaps associated with the vacancy of blocks of a snapshot, and therefore ST1 and ST2 are not snapmaps. They are stripes of file data, each of the stripes containing less data than a file system or a snapshot. It follows that stripe ST10, which integrates ST1 and ST3, is not a *summary map*, but a stripe of file data, and less data than would be typically contained in a file system or in a snapshot. Even if Sekido disclosed a bitmap of ST10, it would not be a bitmap of snapshots, because neither ST1 nor ST3 is a snapshot. At least for these reasons, Sekido does not anticipate independent claims 25, 32, 35, 36, and 37.

Turning next to independent claim 38, Applicants have previously argued that Sekido fails to teach maintaining two active maps, with one of the active maps reflecting fewer free blocks than the other active map. In response, the Final Office Action refers once again to Sekido's Figure 31 and stripes ST1 and ST3. As explained in the preceding paragraphs, ST1 and ST3 are stripes of file data, not bitmaps. Furthermore, Sekido does not teach updating these stripes, but rather repacking them, *i.e.*, combining them into a single stripe. At least for this reason, claim 38 is not anticipated by Sekido.

Claims 36, 38, and 39 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Sekido in view of Rungta. In particular, the Final Office Action stated that Rungta teaches the limitations relating to *consistency point*. The term *consistency point* is expressly defined in the present application:

> • Consistency Point (CP) — In general, the term "CP" refers to a time that a file system reaches a consistent state. When this state is reached, all the files have been written to all the blocks and are safely on disk and the one or more copies of redundant fsinfo blocks get written out. If the system crashes before the fsinfo blocks go out, all other changes are lost and the system reverts back to the last CP. The file system advances atomically from one CP to the next.

Specification, at page 9, line 18 through page 10, line 2. A consistency point recited in the claims thus requires that all files have been written to and the entire file system be in a consistent state. In contrast, Rungta describes "snapshots" and "consistency" that relate to individual files, not to file system as a whole. Consider, for example, step 310 in Rungta's Figure 3 ("Create a snapshot . . . for the open file"). Consider also the following statements:

- "Whenever a file is opened for writing, a snapshot of the file is created. The snapshot includes a copy of the file's metadata." Rungta, the Abstract; *id.*, col. 1, lines 42-44 (emphasis added).

- "Since snapshots are created as files are opened for writing, only one snapshot exists for a file, regardless of the number of times the file is simultaneously open for writing. Only one snapshot is necessary, as it allows the archive unit to archive the most recent consistent version of the file." Rungta, col. 3, lines 13-17.

Rungta's references to "consistency" are similarly made in the context of individual files, not file system as a whole. See, for example, Rungta, col. 2, lines 17-19 and 34-36; *id.* col. 3, lines 15-17 and 28-31; and *id.* col. 4, lines 2-5, 19-21, 35-38, 47-50, 54-55, and 58-60. In each of these cited portions, Rungta refers to file consistency, not to file system consistency. Rungta therefore does not disclose consistency point as defined in the present application. At least for this reason, claims 36, 38, and 39 are patentable over Sekido and Rungta.

The above discussion addresses rejection of all independent claims and of some dependent claims. As regards the dependent claims not specifically discussed, these claims are patentable together with their base claims and intervening claims, if any.

CONCLUSION

For the foregoing reasons, Applicants respectfully submit that all pending claims are patentable over Sekido and Rungta. To discuss any matter pertaining to the present application, the Examiner is invited to call the undersigned attorney at (858) 720-9431.

Having made an effort to bring the application in condition for allowance, a timely notice to this effect is earnestly solicited.

Respectfully submitted,

Dated: May 17, 2005

Anatoly S. Weiser
Reg. No. 43,229

The Swernofsky Law Group
P.O. Box 390013
Mountain View, CA 94039-0013
(650) 947-0700